Kernel Benchmarks and Metrics for Polymorphous Computer Architectures

James Lebak
Hank Hoffmann
Janice McMahon
MIT Lincoln Laboratory

Polymorphous computer architectures (PCA) are new computer architectures being developed under a DARPA/IPTO program to support mission agility for future high performance DoD embedded applications. These new architectures will have the ability to “morph” into different modes of execution with the goal of delivering uniform, high performance across a large variety of different processing types and workload compositions. Examples of these architectures include the MIT RAW machine [5], the Stanford Smart memories project [3], and the University of Texas TRIPS machine [4].

To evaluate the applicability of PCA to next generation ISR (intelligence, surveillance, reconnaissance) applications, MIT Lincoln Laboratory has developed example applications and kernel benchmarks that span the space of embedded ISR application requirements. Matlab code for an example ISR application, with elements of feature-aided tracking [6], is being analyzed by teams developing PCA architectures. In addition, seven kernel benchmarks that represent important pieces of this application have been defined. These seven kernels are FIR filter, singular value decomposition, constant false-alarm rate (CFAR) detection, corner turn, pattern matching, graph optimization via genetic algorithm, and database search.

An important first step in evaluating PCA architectures is the implementation of these kernel benchmarks on processors used in modern embedded applications. This implementation provides a baseline for future comparisons. MIT/LL has implemented these seven kernels on the PowerPC G4 processor. The results show that the throughput varies considerably from kernel to kernel. This variation in performance is reflected in a metric known as *stability*. Defined by Kuck [2], stability is the ratio of the minimum to the maximum throughput for a particular set of problems. A chief benefit of PCA architectures is expected to be their stable performance across a range of kernels and data sizes.

Hoffman [1] has implemented convolution and many other kernels on the RAW simulator using scalable systolic algorithms. Hoffmann’s throughput results for real convolution on a simulated 250 MHz RAW are shown in Figure 1 and compared with a similar kernel on a 500 MHz G4. Both machines have a peak throughput rated at 4 Gflops/sec. Clearly, the simulation results show that RAW has the potential to perform much better than the G4 on this kernel.

In this talk, we present and analyze performance results for several PCA kernels on the MIT RAW simulator and on a RAW test board. We compare these with the baseline performance results obtained on the PowerPC G4 in terms of throughput, stability, efficiency and power efficiency.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **20 AUG 2004**	2. REPORT TYPE **N/A**	3. DATES COVERED **-**
4. TITLE AND SUBTITLE **Kernel Benchmarks and Metrics for Polymorphous Computer Architectures**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **MIT Lincoln Laboratory**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

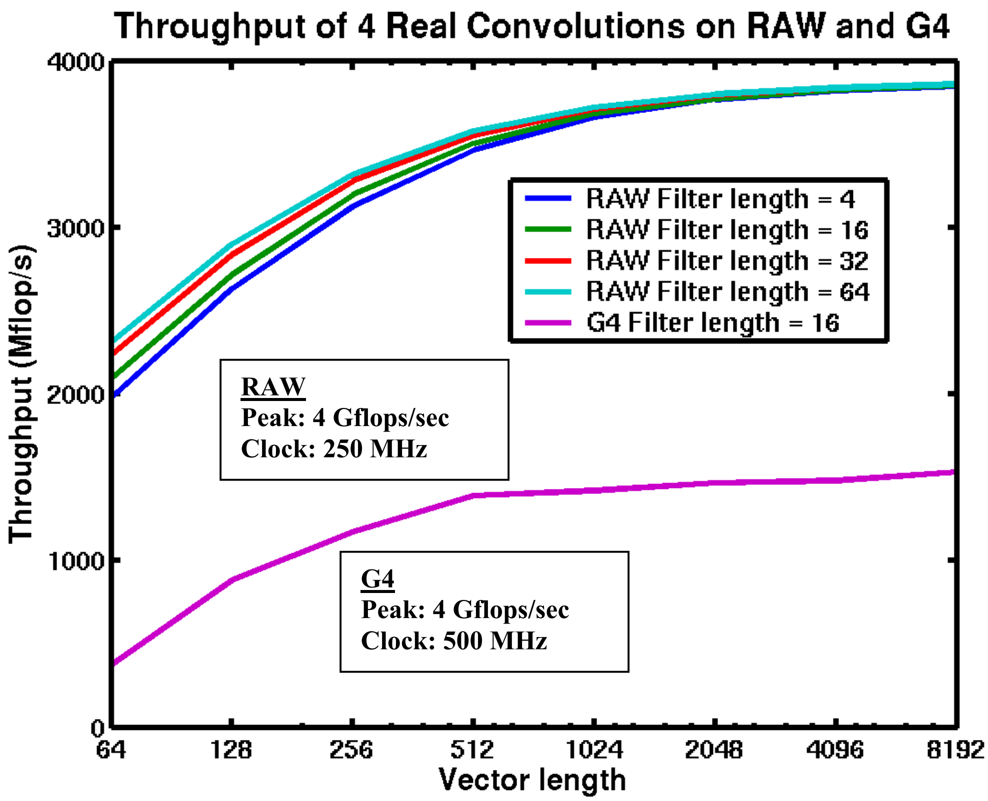
13. SUPPLEMENTARY NOTES
See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**UU**	**28**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18



References:

1. Henry Hoffmann, Volker Strumpen, and Anant Agarwal. Stream Algorithms and Architectures. Technical memo MIT-LCS-TM-636, MIT Laboratory for Computer Science, Cambridge, MA, March 2003.
2. David J. Kuck. *High Performance Computing: Challenges for Future Systems*. New York: Oxford University Press, 1996.
3. Ken Mai, Tim Paaske, Nuwan Jayasena, Ron Ho, William J. Dally, and Mark Horowitz. Smart Memories: A Modular Reconfigurable Architecture. In *28th Annual International Symposium on Computer Architecture*, pages 161–171, June 2000.
4. Ramdass Nagarajan, Karthikeyan Sankaralingam, Doug C. Burger, and Steve W. Keckler. A Design Space Evaluation of Grid Processor Architectures. In *34th Annual International Symposium on Microarchitecture*, pages 40–51, December 2001.
5. Michael B. Taylor, Jason Kim, Jason Miller, David Wentzlaff, Fae Ghodrat, Ben Greenwald, Henry Hoffmann, Paul Johnson, Jae-Wook Lee, Walter Lee, Albert Ma, Arvind Saraf, Mark Seneski, Nathan Shnidman, Volker Strumpen, Matt Frank, Saman Amarasinghe, and Anant Agarwal. The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs. *IEEE Micro*, 22(2):25–36, March/April 2002.
6. Duy H. Nguyen, John H. Kay, Bradley J. Orchard, and Robert H. Whiting. Classification and Tracking of Moving Ground Vehicles. *Lincoln Laboratory Journal*, 13(2):275–308, 2002.

Abstract

Presentation

Back to Agenda

Next Abstract

Kernel Benchmarks and Metrics for Polymorphous Computer Architectures

Hank Hoffmann

James Lebak (Presenter)

Janice McMahon

MIT Lincoln Laboratory

Seventh Annual High-Performance Embedded Computing Workshop (HPEC)

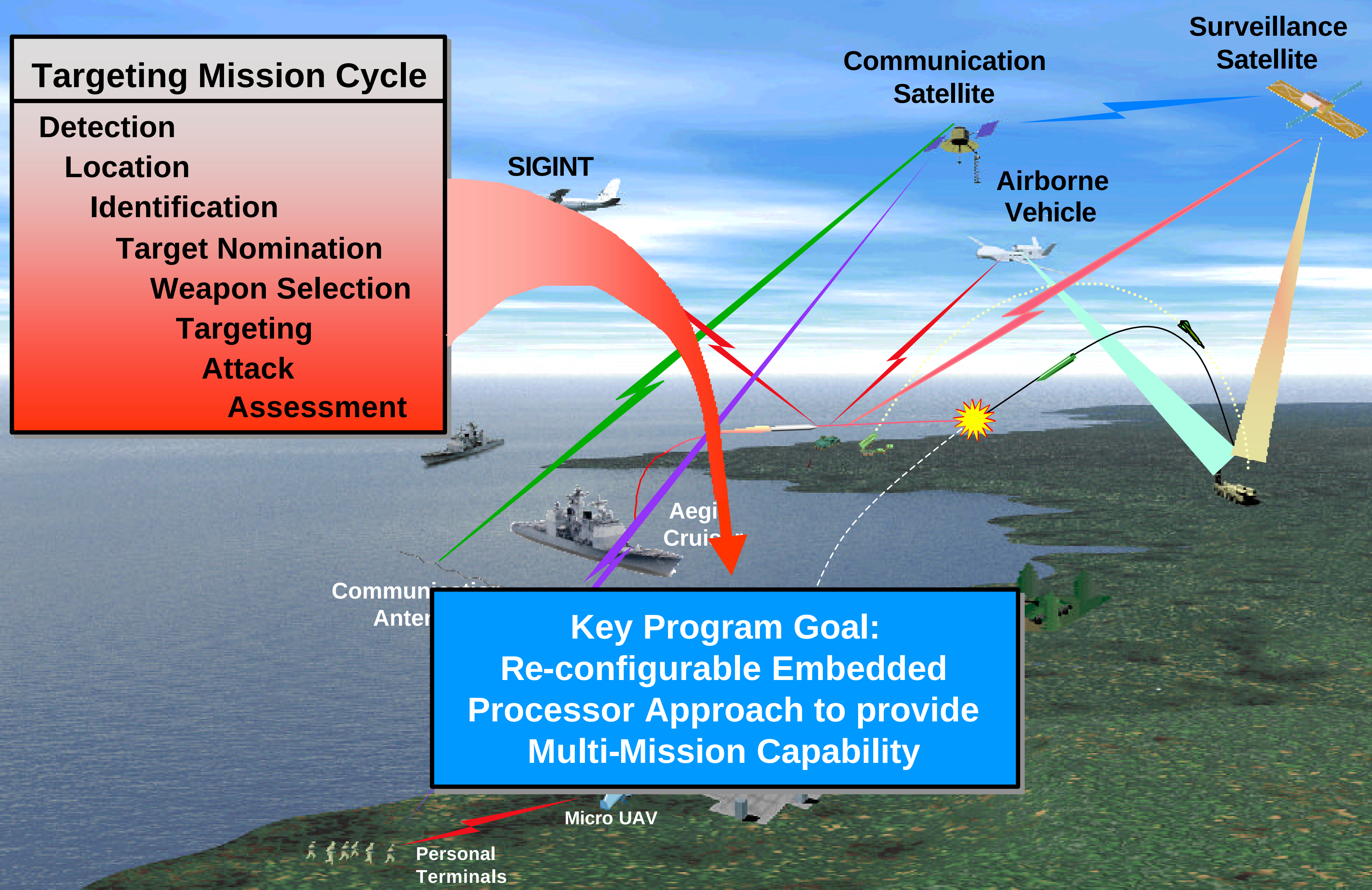
24 September 2003

This work is sponsored by the Defense Advanced Research Projects Agency under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

Future Warfighting Scenarios
Examples

Targeting Mission Cycle

- Detection
- Location
- Identification
- Target Nomination
- Weapon Selection
- Targeting
- Attack
- Assessment



Key Program Goal:
Re-configurable Embedded Processor Approach to provide Multi-Mission Capability

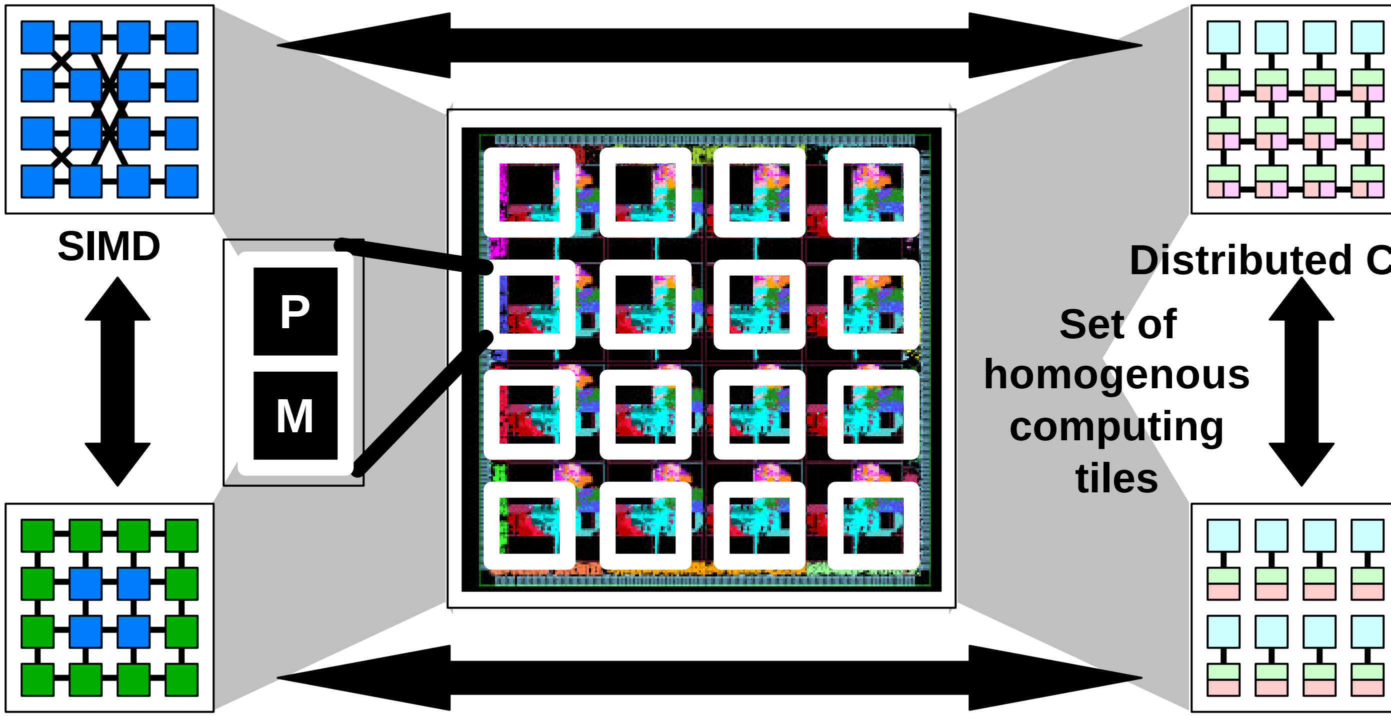
Polymorphous Computing

Stream processing

- Regular, deterministic operations
- Constant flow of input data

Threaded processing

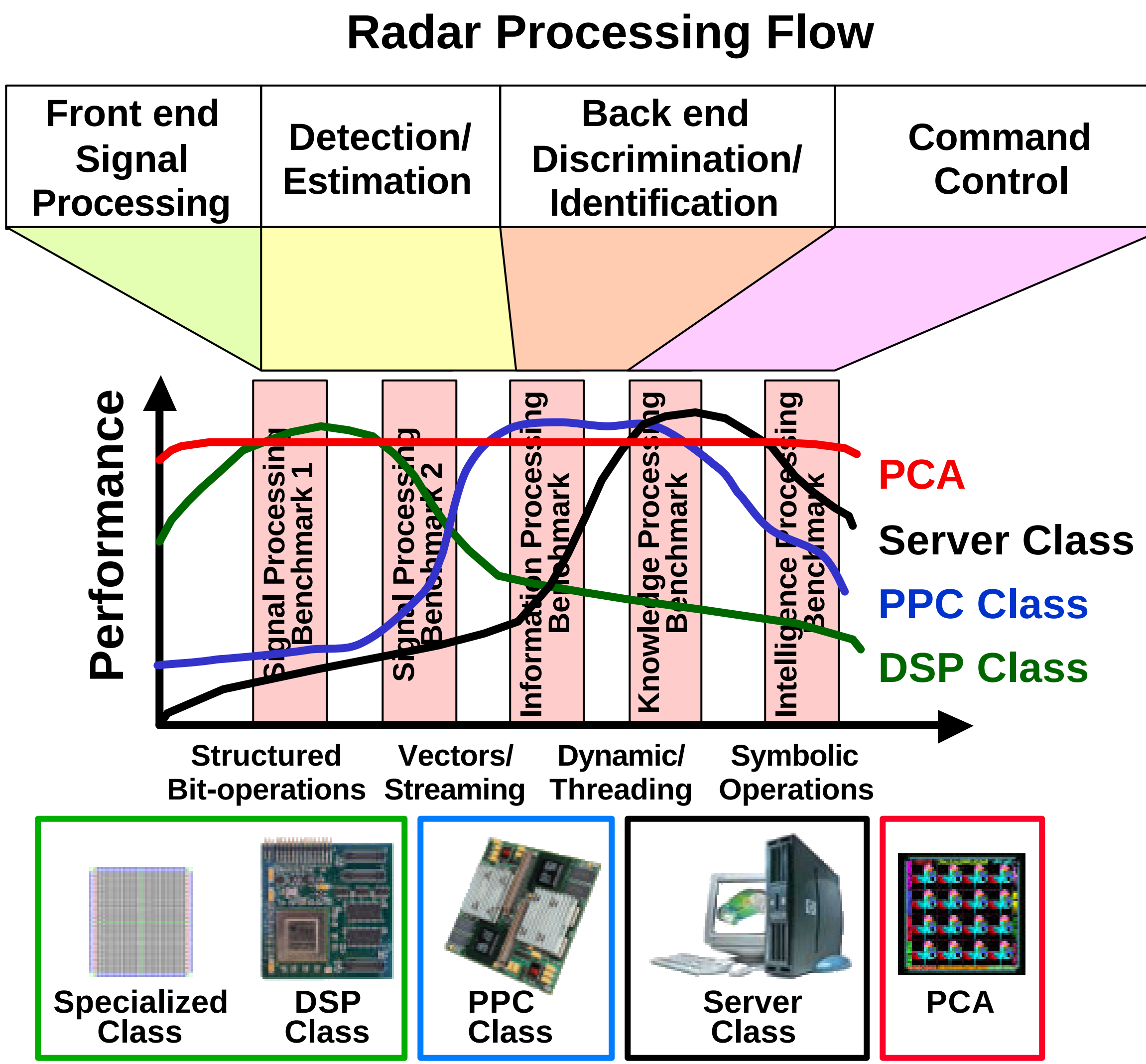
- Complex operations
- Dynamic data movement



[1]**morph** \'mor-()f\ *n* : re-structuring of tiles for optimized processing

[2]**morph** \'mor-()f\ *vt* : to re-structure tiles for optimized processing

Architectural Flexibility



Outline

- Introduction
- Kernel Benchmarks and Metrics
- Programming PCA Architectures
- Case Study: SVD Kernel
- Conclusions

Kernel Synthesis from Application Survey

Specific Application Areas

Radar | Sonar | Infrared | Hyper-Spectral | SIGINT | Communication | Data Fusion

Broad Processing Categories

"Front-end Processing"

- Data independent, stream-oriented
- Signal processing, image processing, high-speed network communication
- Examples:
  - pulse compression
  - adaptive beamforming
  - target detection

"Back-end Processing"

- Data dependent, thread oriented
- Information processing, knowledge processing
- Examples:
  - workload optimization
  - target classification

Specific Kernels

Signal/Image Processing	Communication	Information/Knowledge Processing
• FIR Filter • SVD • CFAR Detection	• Corner Turn	• Graph Optimization • Pattern Recognition • Real-time Database Operations

MIT-LL Surveyed DoD Applications to Provide:

- Kernel Benchmark Definitions
- Example Requirements and Data Sets

Kernel Performance Evaluation

Kernel Benchmarks

Signal/Image Processing
- FIR Filter
- SVD
- CFAR Detection

Communication
- Corner Turn

Information/Knowledge Processing
- Graph Optimization
- Pattern Recognition
- Real-time Database Operations

Performance Metrics

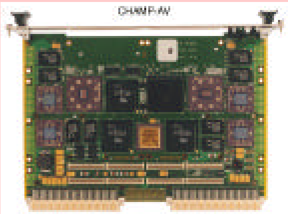
- Floating point and integer ops
- Latency
- Throughput
- Efficiency
- Stability
- Density and cost
  - –Size
  - –Weight
  - –Power

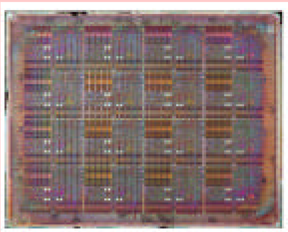
Definitions

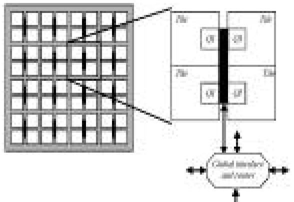
$$\text{Throughput} = \frac{\text{Workload (FLOPS or OPS)}}{\text{Execution time (seconds)}}$$

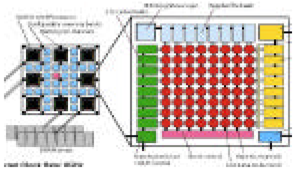
$$\text{Efficiency} = \frac{\text{Throughput}}{\text{Hardware Peak}}$$

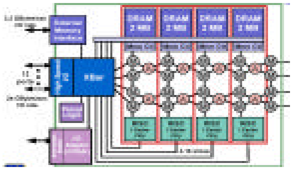
$$\text{Stability} = \frac{\text{MIN(Throughput)}}{\text{MAX(Throughput)}}$$


PowerPC(G4)


RAW


Smart Memory


TRIPS

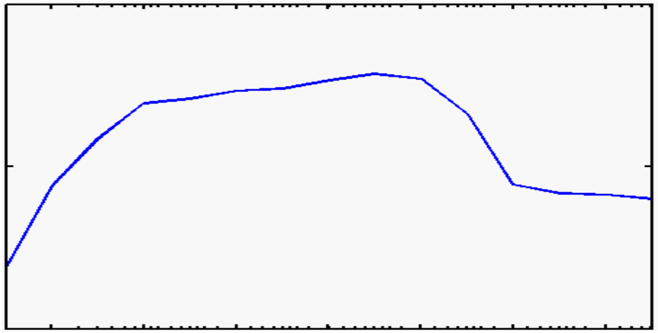

MONARCH

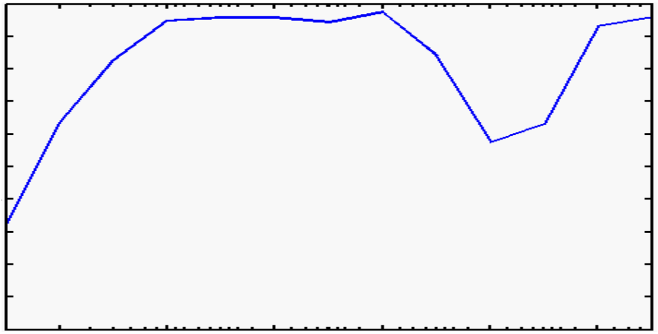
Throughput-Stability Product

A New Kernel Metric

Throughput

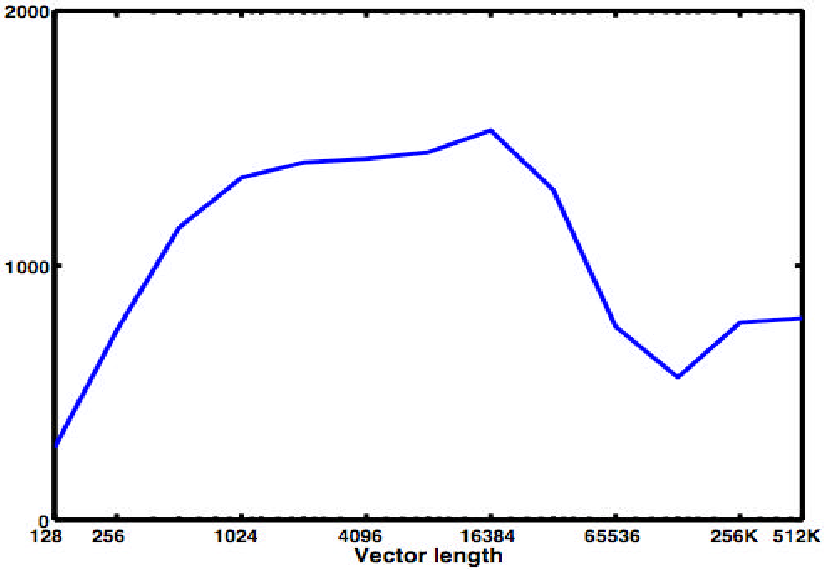
$$\frac{\text{Workload (FLOPS or OPS)}}{\text{Execution time (seconds)}}$$





Interval Stability

$$\frac{\text{MIN}_I(\text{Throughput})}{\text{MAX}_I(\text{Throughput})}$$



Throughput x *Stability*

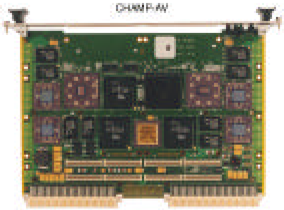
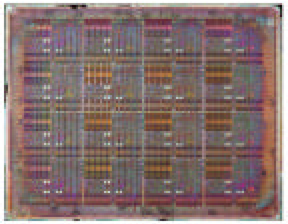
- rewards consistent high performance
- penalizes lack of performance or lack of consistency

For a given application, PCA processors should achieve higher product of throughput and stability than conventional processors

Outline

- Introduction
- Kernel Benchmarks and Metrics
- Programming PCA Architectures
- Case Study: SVD Kernel
- Conclusions

High Performance Programming: Conventional vs. PCA Processors

PowerPC(G4)	Raw
	

Characteristics:
- Rigid memory hierarchy
- Rigid datapath
- Specialized Structures

High Performance Programming:
- Change algorithm to match memory hierarchy
- One degree of freedom
- Can only work with blocking factor

Characteristics:
- Flexible memory hierarchy
- Flexible datapath(s)
- Generic Structures

High Performance Programming:
- Co-optimize algorithm and architecture
- Many degrees of freedom
- Optimize time/space tradeoff

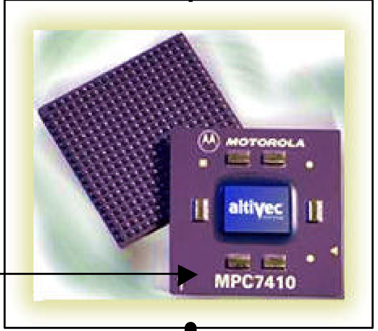
PCA provides more degrees of freedom, and thus greater flexibility (morphability) and greater performance over a range of applications

Kernel Benchmarks and the PowerPC G4

Main Memory

PowerPC G4 7410 Specs

- 500 MHz Clock rate
- 4 Gflop/s peak
- 125 MHz main memory bus
- L1 cache: 32 kB, on chip
- L2 cache: 2MB, 250 MHz bus
- Mercury daughtercard

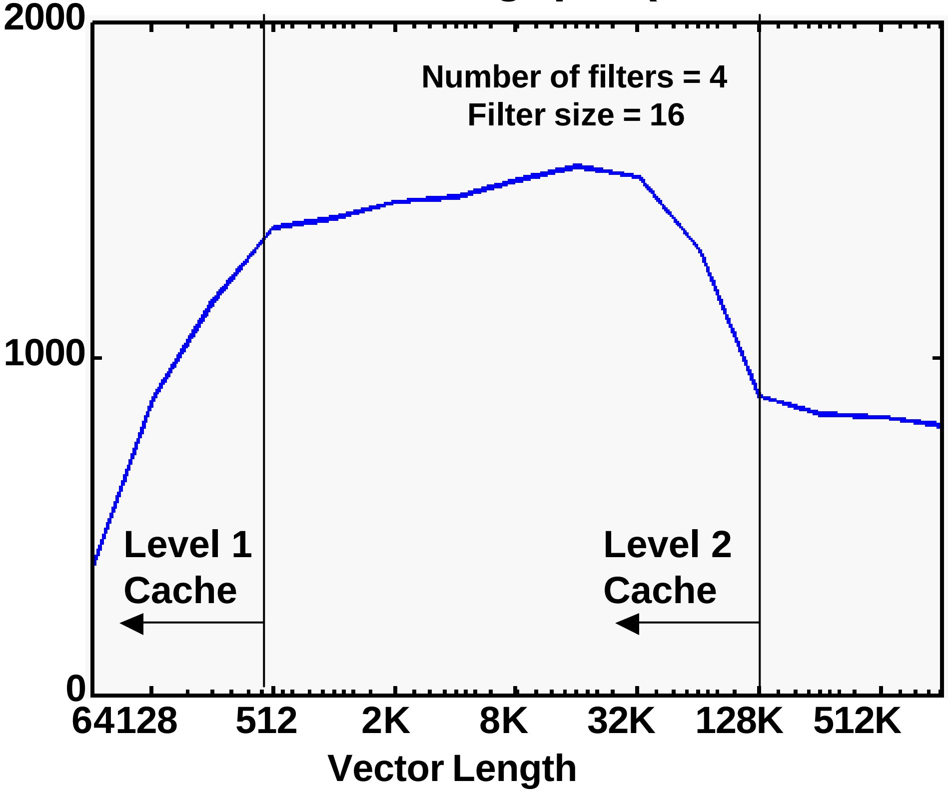


L2 Cache

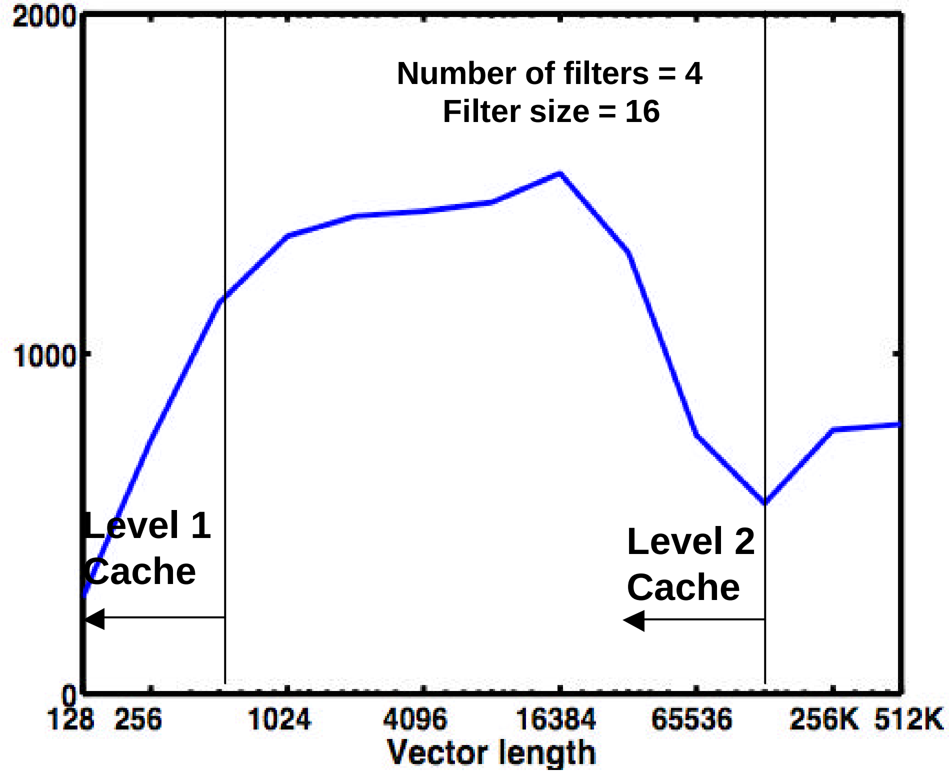
- **Two predictors of kernel performance:**
  - Programmer’s maximization of data reuse and locality (blocking factor)
  - Memory hierarchy of G4
- **Blocking factor determines max achieved performance**
- **Memory hierarchy determines shape of performance curve**
- **Want to maximize blocking factor to limit memory hierarchy bottleneck**

FIR Filter (G4)

FIR Filter Throughput (MFLOPS/sec)



FIR Throughput ? Stability



*Implemented with VSIPL Real FIR Filter

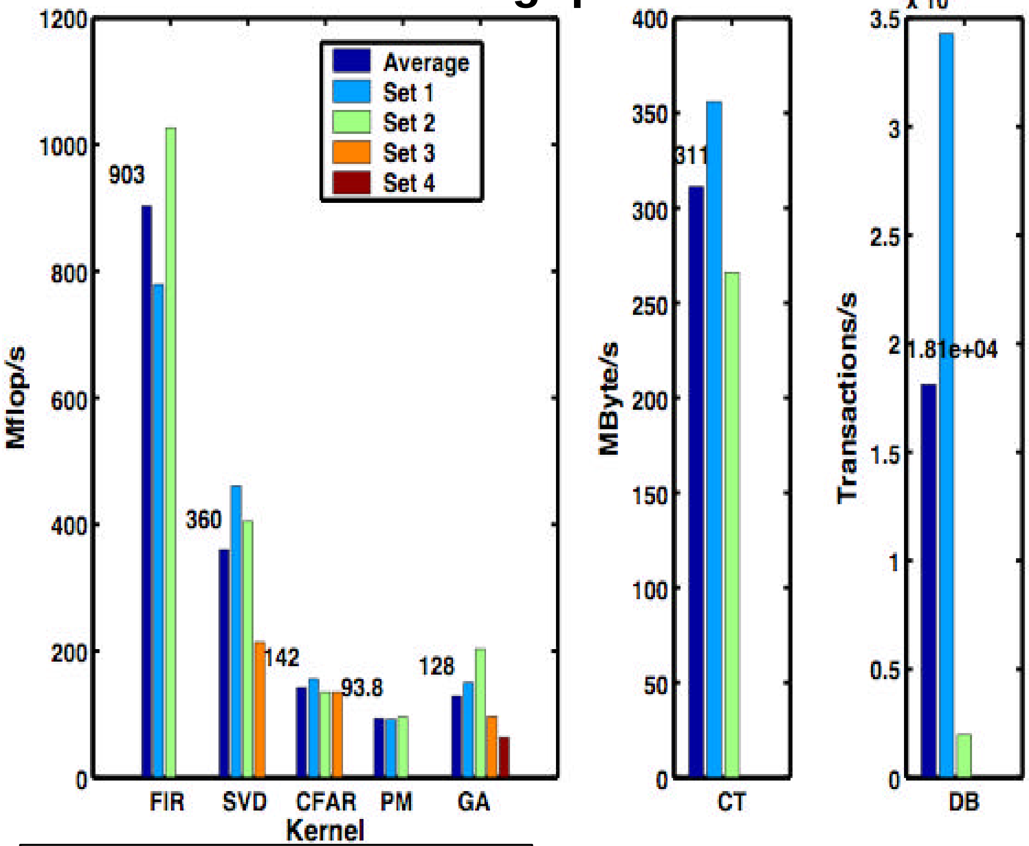
PowerPC G4 (Mercury)
- 500 MHz
- Peak: 4 GFLOPS/sec

Mean Efficiency: 29%

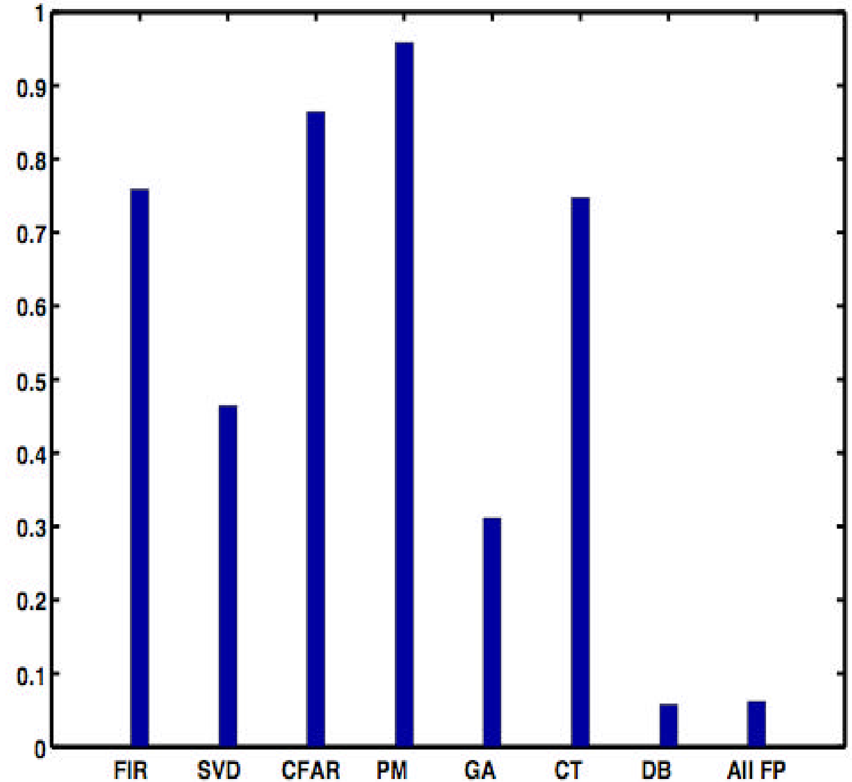
Caches are performance bottlenecks
- Performance curve changes when cache is full
- Product metric penalizes G4 for performance drop at cache boundaries

Baseline Performance Measurements: Throughput and Stability

Throughput



Data Set and Overall Stability



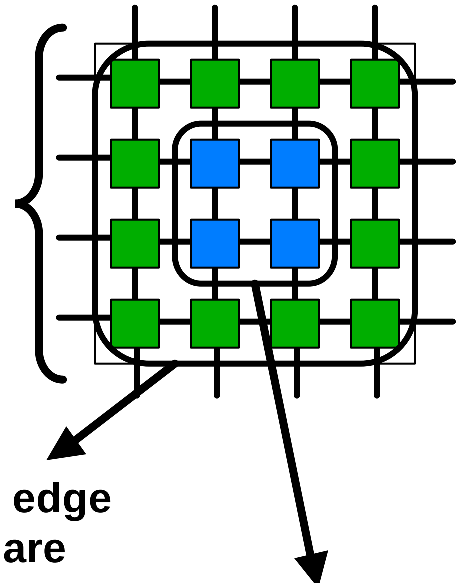
PowerPC G4 (Mercury)
- 500 MHz
- 32 KB L1
- 2 MB L2
- Peak: 4 GFLOPS/sec

Data Set Stability:	Ratio of minimum to maximum over all data set sizes for a particular kernel
Overall Stability:	Ratio of minimum to maximum over all floating-point kernels&all data set sizes

Stream Algorithms for Tiled Architectures

Systolic Morph

R



M(R) **edge tiles are allocated to memory management**

P(R) **inner tiles perform computation systolically using registers and static network**

Time **Space**

Stream Algorithm Efficiency:

$$E\,(N,R) = \frac{C(N)}{T(N,R)*(P(R) + M(R))}$$

where

N = problem size
R = edge length of tile array
C(N) = number of operations
T(N,R) = number of time steps
P(R) + M(R) = total number of processors

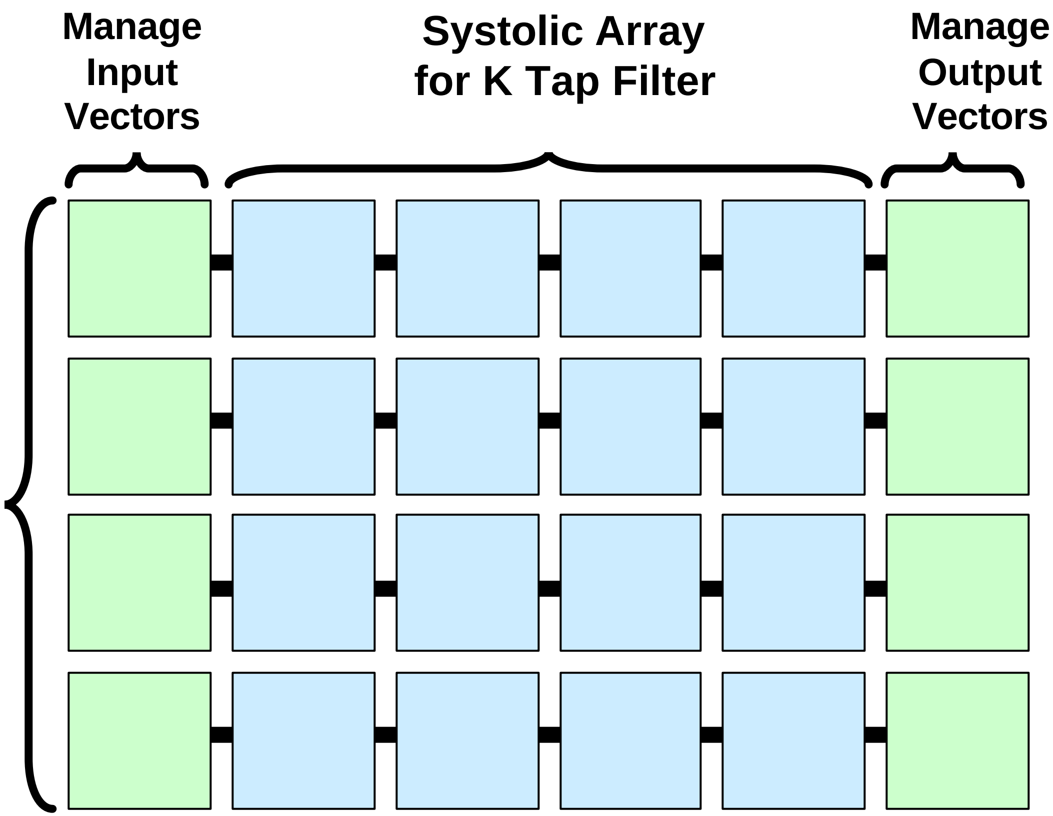
Compute Efficiency Condition: $\lim_{?,R\,?\;?} E(?,R) = 1$

where ? = N/R

Stream algorithms achieve high efficiency by optimizing time space tradeoff – tailoring memory hierarchy and datapaths to specific needs of application

Time Domain Convolution on RAW

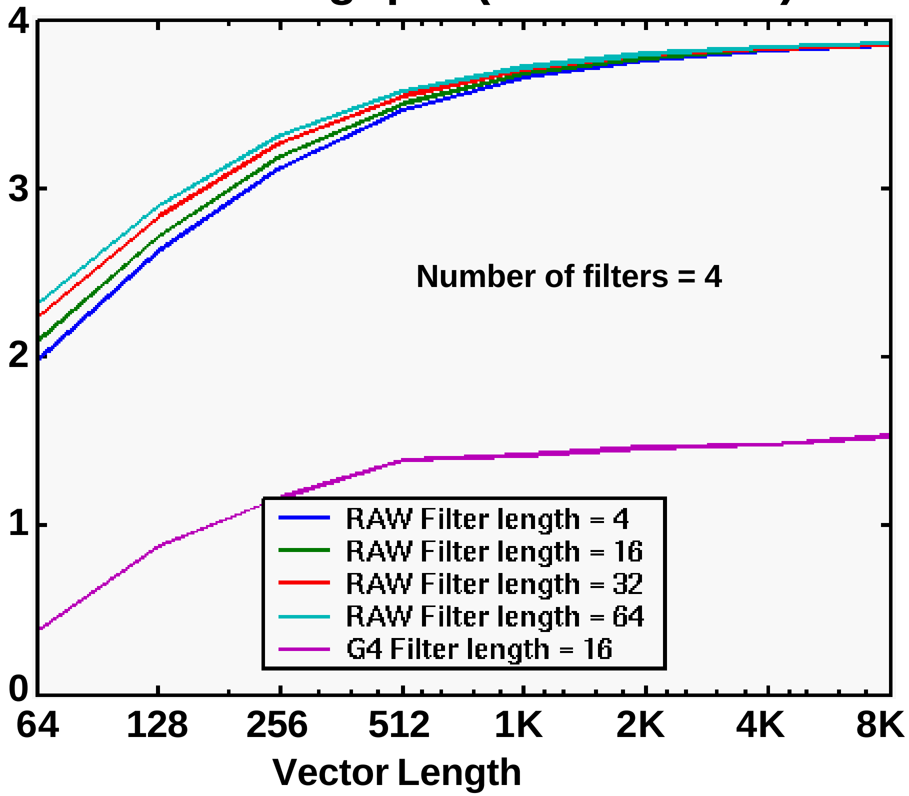
RAW Chip with R rows and
R+2 columns:
Number of filters = R
Number of memory tiles:
$M = 2*R$
Number of processing tiles:
$P = R^2$



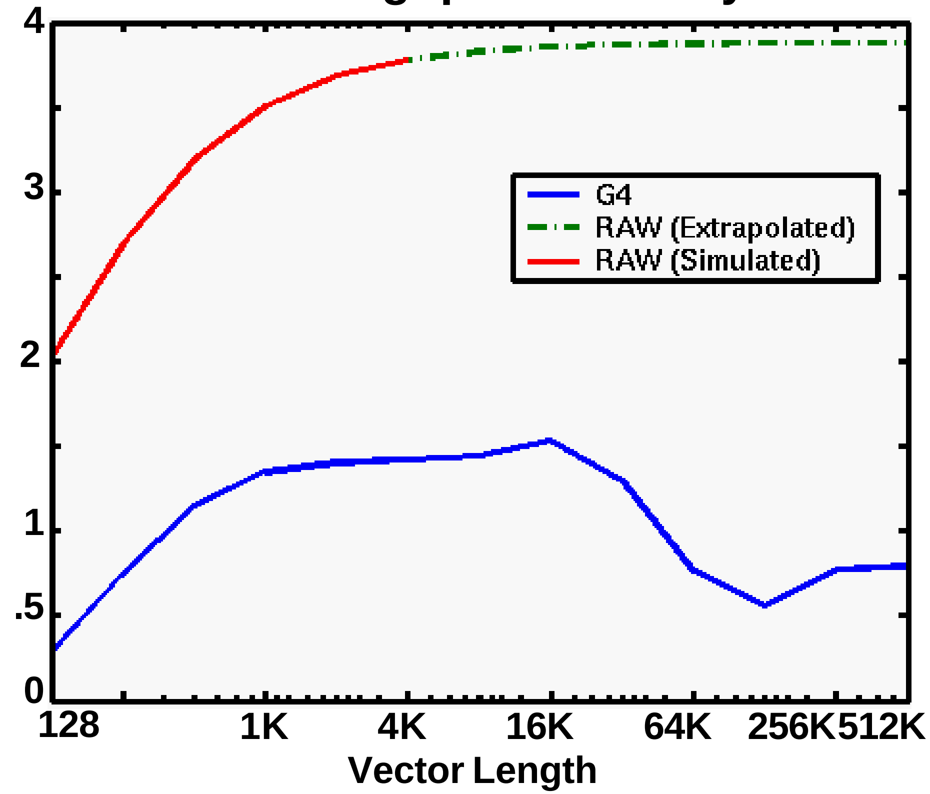
Stream algorithms achieve high performance by removing memory access bottleneck from computational critical path

FIR Filter (RAW)

Throughput (GFLOPS/sec)



Throughput * Stability



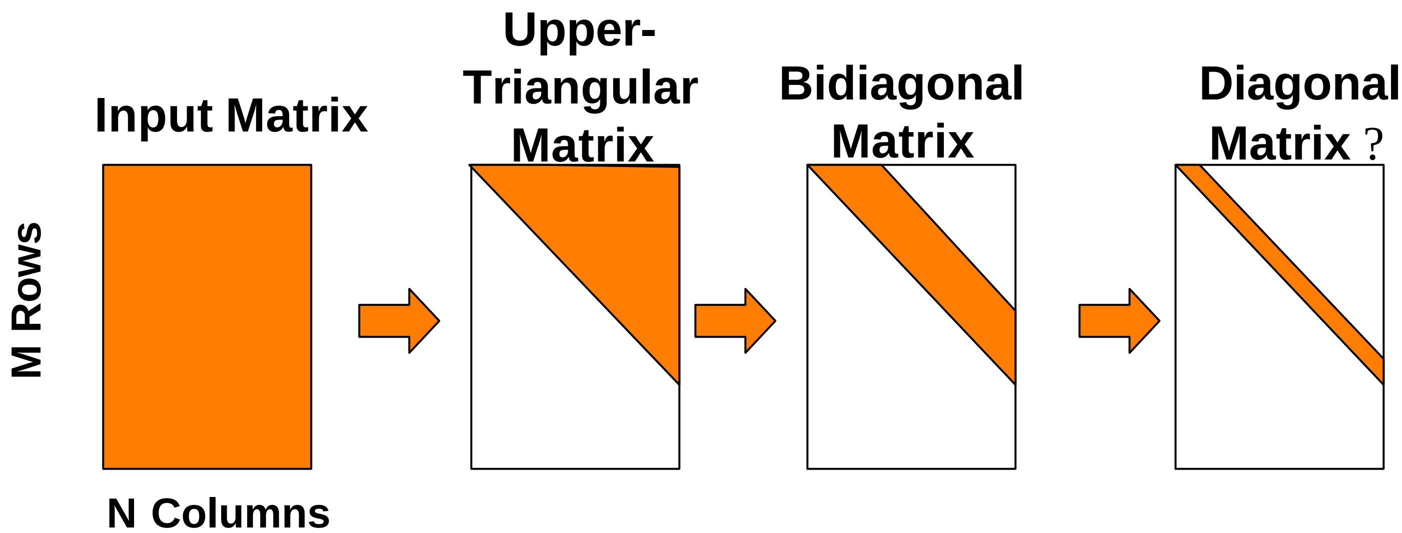
RAW: 250 MHz, 4 GFLOPS/sec	G4: 500 MHz, 4 GFLOPS/sec

Raw implements the appropriate memory hierarchy for the problem
Raw's Throughput x Stability score stays high

Outline

- Introduction
- Kernel Benchmarks and Metrics
- Programming PCA Architectures
- Case Study: SVD Kernel
- Conclusions

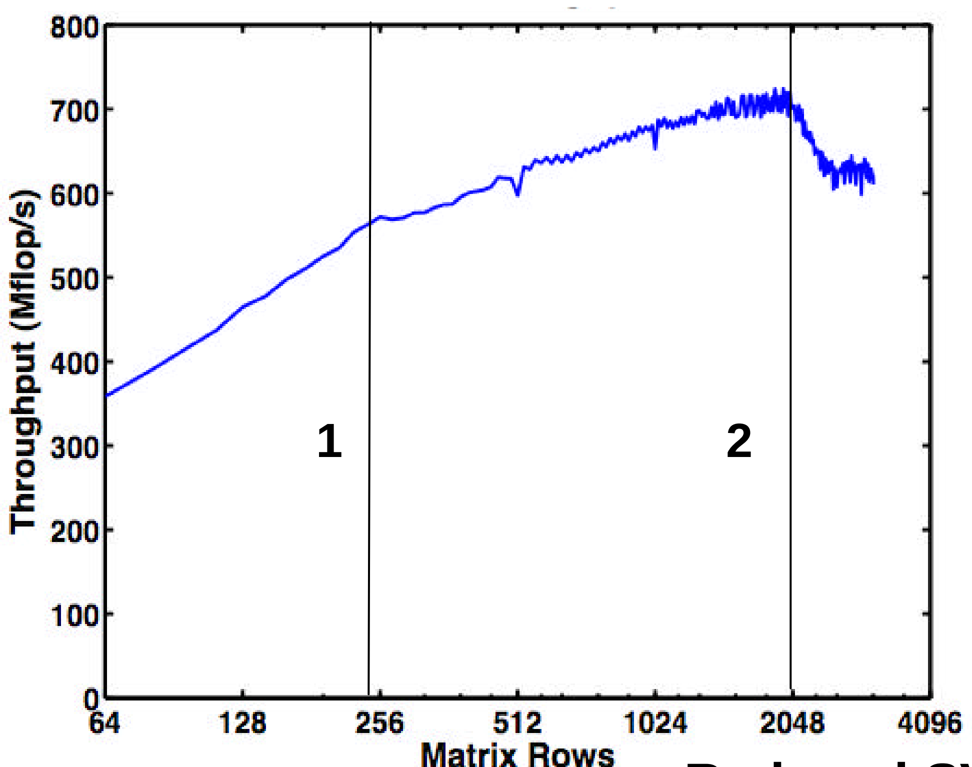
Singular Value Decomposition (SVD)

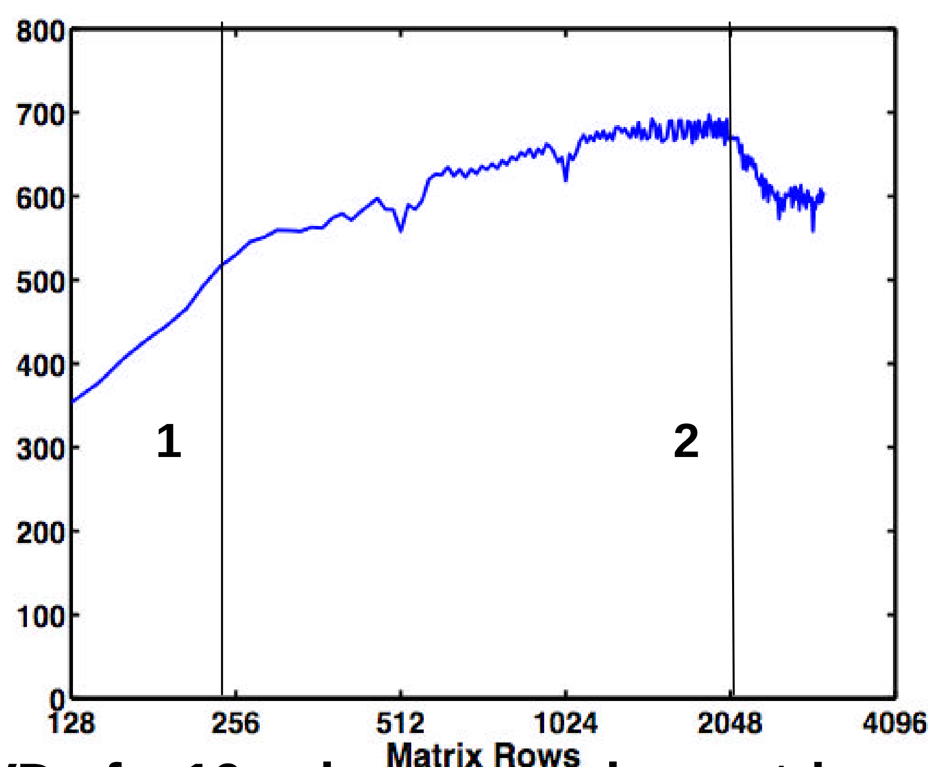


X=U? V^H
U, V Unitary
? ?Diagonal

- SVD is becoming more widely used in signal and image processing
  - Important for spectral analysis
  - Can also be used for adaptive beamforming, especially for ill-conditioned problems
- SVD kernel implementation is a Reduced SVD that begins with a QR factorization if M > N
  - Uses Modified Gram-Schmidt QR factorization
  - Many possible optimizations, especially block factorization

SVD Results (G4)





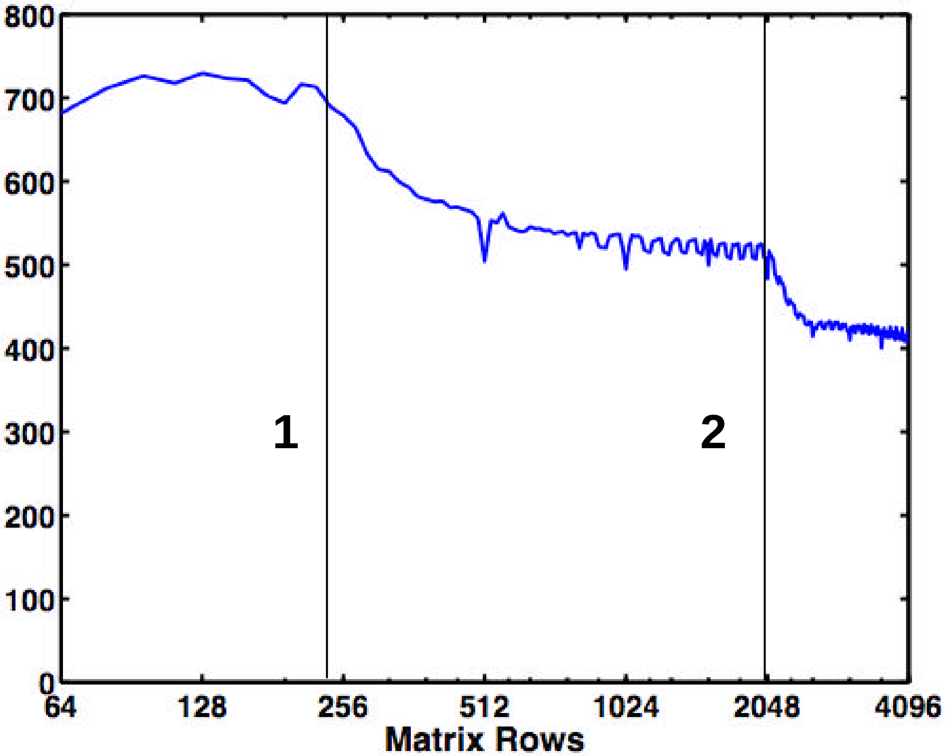
- Reduced SVD of a 16-column complex matrix
- Begins with MGS QR factorization (needs A+R)
- L1 cache drives inner loop performance
  - 1: A+R fills L1 cache
  - 2: One column of A is half of L1 cache

PowerPC G4 (Mercury)
- 500 MHz
- Peak: 4 GFLOPS/sec

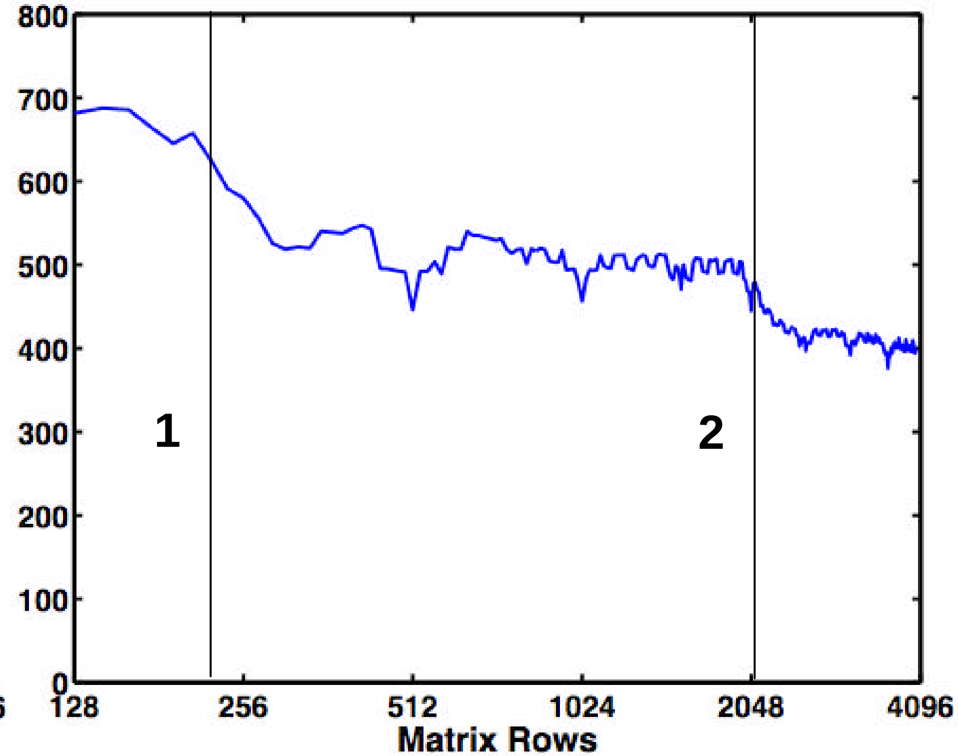
Mean Efficiency: 16%

Modified Gram-Schmidt QR Results (G4)

MGS Throughput (Mflop/s)



MGS Throughput ? Stability

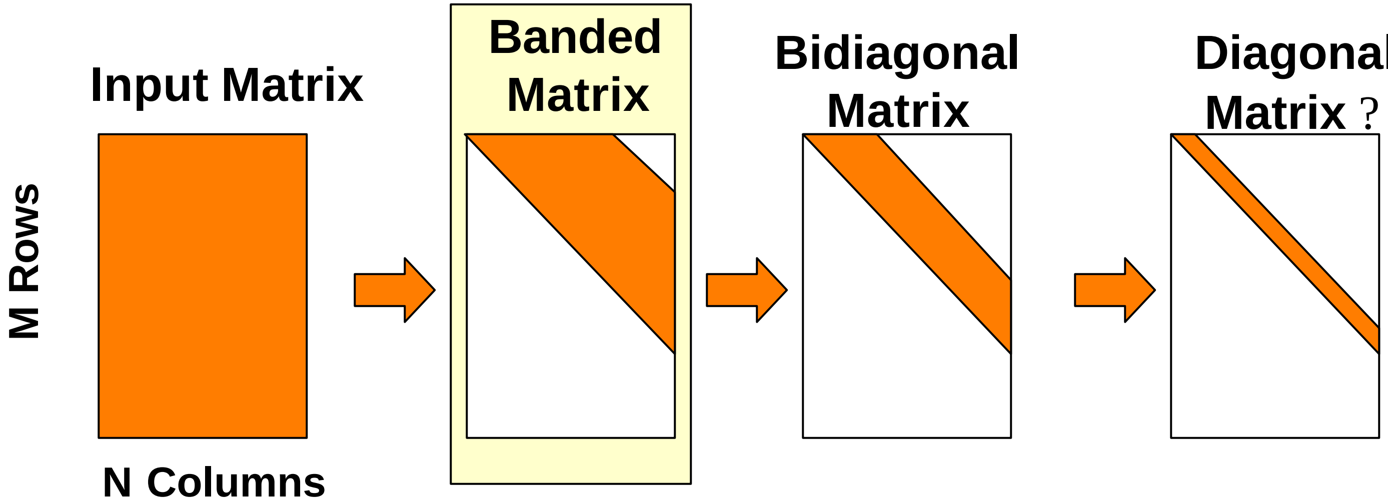


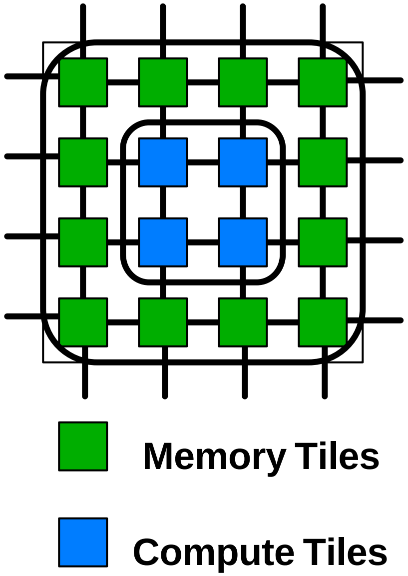
PowerPC G4 (Mercury)
- 500 MHz
- Peak: 4 GFLOPS/sec

Mean Efficiency: 12%

- Modified Gram-Schmidt QR factorization of a 16-column complex matrix
- MGS is about 60% of SVD time
- L1 cache drives inner loop performance
  - 1: A+R fills L1 cache
  - 2: One column of A is half of L1 cache

SVD for RAW Architecture



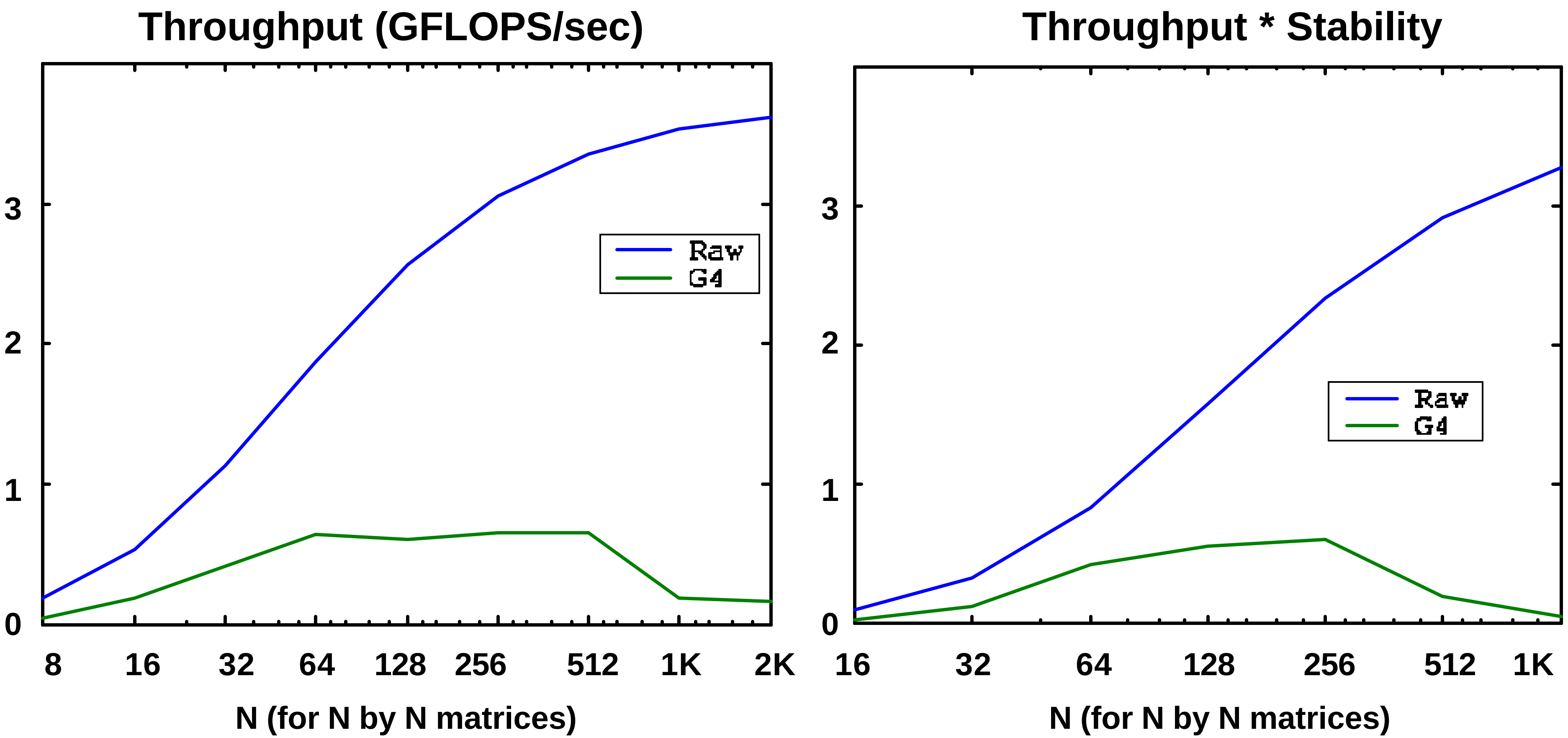


- Goal is to match problem size and architecture
- Use 2D systolic morph
  - maximizes time/space efficiency
  - uses architecture in a scalable way
- Uses efficient QR/LQ approach to get to banded form
  - Fast Givens approach for QR/LQ
  - Decoupled algorithm with good parallelism
- Banded form matches array dimension of systolic morph
  - provides high locality for reduction to bidiagonal form

Raw implementation seeks to efficiently match the many possible algorithms to the many possible architectural configurations

RAW and G4 Results: Fast Givens QR Factorization

The QR is a key sub-kernel of the SVD



The QR performance demonstrates the benefit of the PCA approach on matrix algebra operations

Lincoln Laboratory PCA Testbed

Test Bed Architecture

Clusters
on LLAN

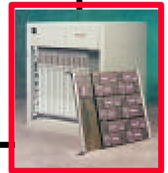
Ethernet LAN

Mercury RACE/VME
Solaris/MCOS

SBC

G4

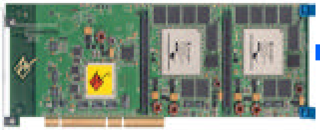
DSP



Intel PC
- Dual processor
- 66 MHz/64-bit wide PCI bus
- Running Linux

PCI bus

Annapolis
Wildstar



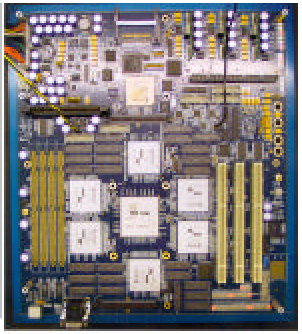
DSP/ FPGA

Unit
under
test

High
Speed
I/O

Test Bed Objectives

- Kernel performance evaluation
- Application morphing demonstration
- High-level software prototyping



RAW Test Board
(October 2003)

- 2 MB DRAM
- High Speed I/O
- USB Interface
- Daughtercard
- High Speed A/D

Outline

- Introduction
- Kernel Benchmarks and Metrics
- Programming PCA Architectures
- Case Study: SVD Kernel
- Conclusions

Conclusions

- **MIT Lincoln Laboratory has defined kernel benchmarks for the PCA program**
  - Multiple categories of processing
  - Based on DoD application needs
- **Establishing a performance baseline on conventional architectures**
  - Performance is limited by the blocking factor and by the memory hierarchy
  - Example: CFAR – low ops/byte, 3% efficiency: FIR – high ops/byte, 29% efficiency
- **PCA processors allow opportunities for high performance**
  - Performance achieved through co-optimization of the algorithm and the architecture
  - Example: unusual SVD algorithm leads to high performance on Raw
  - The greater degree of freedom allows greater optimization across a variety of problem domains

MIT Lincoln Laboratory PCA Team

Abstract Presentation

Back to Agenda

Next Abstract

Hector Chan
Bill Coate
Jim Daly
Ryan Haney
Hank Hoffmann
Preston Jackson
James Lebak
Janice McMahon
Eddie Rutledge
Glenn Schrader
Edmund Wong